Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)
Vol. 2, No. 2, Oktober 2015, hlm. 87-95

PENGKLASIFIKASIAN DOKUMEN BERBAHASA INDONESIA DENGAN PENGINDEKSAN BERBASIS LSI

Achmad Ridok[1], Indriati [2]

[1,2]Dosen pada Fakultas Ilmu Komputer Unibraw
Email: [1]acridokb@ub.ac.id, [2]indtriati@ub.ac.id

(Naskah masuk: 11 Juni 2015, diterima untuk diterbitkan: 22 Juli 2015)

Abstrak

Klasifikasi dokumen teks bertujuan untuk menentukan kategori suatu dokumen berdasarkan kesamaannya dengan kumpulan dokumen yang telah berlabel sebelumnya. Namun demikian kebanyakan metode klasifikasi yang ada saat ini dilakukan berdasarkan kata-kata kunci atau kata-kata yang dianggap penting dengan mengasumsikan masing-masing merepresentasikan konsep yang unik. Padahal pada kenyataanya beberapa kata yang mempunyai makna atau semantik sama seharusnya diwakili satu kata unik. Pada penelitian ini pendekatan berbasis LSI (*Latent Semantic Indexing)* digunakan pada KNN untuk mengklasifikasi dokumen berbahasa Indonesia. Pembobotan term dari dokumen-dokumen latih maupun uji menggunakan tf-idf, yang direpresentasikan masing-masing dalam matrik term-dokumen A dan B. Selanjutnya matrik A didekomposisi menggunakan SVD untuk mendapatkan matrik U dan V yang tereduksi dengan k-rank. Kedua matrik U dan V digunakan untuk mereduksi B sebagai representasi dokumen uji. Evaluasi kinerja sistem terbaik berdasarkan hasil diperoleh pada klasifikasi KNN berbasis LSI tanpa stemming dengan *threshould* 2. Akan tetapi evaluasi kinerja terbaik berdasarkan waktu dicapai ketika KNN LSI dengan stemming pada *threshould* 5. Kinerja KNN berbasis LSI secara signifikan jauh lebih baik dibandingkan dengan KNN biasa baik dari sisi hasil maupun waktu.

Kata kunci: *KNN, LSI, K-Rank, SVD, Klasifikasi dokumen*

Abstract

Classification of text documents aimed to determine the category of a document based on its similarity to set of documents which have been previously labeled. However, most existing methods of classification were conducted based on key words or words that are considered important by assuming each representing a unique concept. Whereas in fact some of the words that have the same meaning or semantics should be represented as a unique word. In this research LSI -based approach used on KNN to classify documents in Indonesian language. Weighting the terms of the training documents or testing using tf-idf, which represented respectively in term-document matrix A and B. Furthermore, the matrix A is decomposed using SVD to obtain matrices U and V are reduced by k-rank. Both matrices U and V are used to reduce B as a representation of test documents. The best system performance evaluation based on the results obtained LSI-based in the KNN classification without stemming with threshould 2. However, the best performance evaluation based on the time achieved when KNN LSI with stemming the KNN with threshould 5. Performance-based LSI is significantly much better than the tradisional KNN in term both the outcome and timing.

Keywords: *KNN, LSI, K-Rank, SVD, Documents classification*

1. PENDAHULUAN

Ketersediaan dokumen teks digital semakin bertambah dari tahun ke tahun sebagai dampak perkembangan teknologi informasi. Hampir semua aktifitas tulis menulis saat ini semuanya telah menggunakan media elektronik baik komputer maupun hp, berupa pengiriman surat elektronik, sms, penulisan artikel, koran digital, majalah digital dan lain sebagainya. Formatnyapun beragam mulai berupa teks biasa, dokumen yang ditulis dengan *Microsoft word*, pdf, html dan XML. Informasi dokumen teks yang demikan besar tersebut perlu dikelola dalam suatu cara sehingga mudah untuk pengaksesan dan penggalian informasi daripadanya. Diantara bentuk pengelolaan dokumen tersebut adalah pengklasifikasi dokumen. Klasifikasi dokumen akan memberi label dokumen yang masih belum mempunyai label berdasarkan kesamaan dengan sekelompok dokumen yang sebelumnya telah diberi label.

Beberapa metode klasifikasi teks telah dikembangkan seperti KNN, *Naïve Bayes*, *Rocchio* dan SVM. Namun demikian kebanyakan metode klasifikasi yang ada saat ini dilakukan berdasarkan kata-kata kunci atau kata-kata yang dianggap penting setelah melalui tahap pra proses. Masing-masing kata tesebut diasumsikan merepresentasikan konsep yang unik. Padahal pada kenyataannya suatu kata dapat mempunyai padanan katanya sehingga dua atau tiga kata yang sebenarnya semakna masing-masing tetap direpresenatasikan sebagai *term* yang unik. Masalah ini dikenal dengan masalah *polysemi* dan *synonymy*. Sebagai akibat dari masalah ini walaupun klasifikasi dapat menghasilkan hasil yang baik, akan tetapi dimensi

term yang terbentuk sangat besar. Representasi *term* dokumen akan membentuk ruang vector *sparse* dengan dimensi yang besar, karena kata-kata yang tidak relevan juga akan diekstrak. Dengan demikian akurasi yang tinggi sulit dicapai selain banyak menghabiskan waktu dalam prosesnya.

Masalah polisemi dan sinonimitas berhubungan dengan skema pengindekan berbasis konseptual. Persoalan ini dapat diselesaikan dengan LSI (*Latent Semantic Indexing*). LSI akan melakukan transformasi ruang vector asal ke ruang *latent semantic*. Dokumen uji dan dokumen latih keduanya direpresentasikan dan dibandingkan dalam ruang vector baru. Pada metode ini data ditransformasi ke dalam suatu ruang konsep yang baru (Silva et al., 2004). Ruang konsep ini tergantung pada koleksi dokumen, karena koleksi yang berbeda akan mempunyai himpunan konsep yang berbeda. Ide dasar LSI adalah memproyeksikan data ke dalam sub ruang dari data asal sehingga pengaruh nois dari sinonimitas dan polisemi dihilangkan.

Walaupun LSI asalnya diusulkan sebagai salah satu metode sistem temu kembali, metode ini juga telah digunakan secara luas dalam pengklasifikasian teks. Sebagai contoh Yang pada (Yang, 1995) menggunakan LSI untuk menghilangkan noise selama proses pelatihan, (Zelikovitz and Hirsh, 2001) melakukan SVD pada satu matrik *term*-dokumen yang diperluas meliputi pelatihan contoh dan latar belakang pengetahuan untuk meningkan klasifikasi, (Wu and Gunopulos, 2002) melakukan SVD pada matrik *term*-dokumen yang melibatkan tem tunggal dan frase.

Atas latar belakang di atas maka pada penelitian ini diekplorasi pengindekan dengan menggunakan LSI pada klasifikasi dokumen berbahasa Indonesia. Metode klasifikasi yang akan dicobakan adalah metode KNN pada berbagai variasi k dan mempertimbangkan nilai ambang frekuensi masing-masing *term.*

Pertimbangan digunakannya metode KNN adalah selain metode ini paling lama dan sederhana dari klasifikasi non parametric, metode ini dalam prosesnya menerapkan konsep pola-pola yang yang berdekatan dalam ruang fitur yang dimasukkan dalam satu kelas yang sama. Klasifikasi suatu pola yang belum diketahui kelasnya akan ditentukan berdasarkan kedekatannya dengan sekumpulan data yang telah mempunyai kelas atau label sebelumnya. Selanjutnya kumpulan dokumen yang telah berlabel ini disebut sebagai dokumen latih dan dokumen yang masih belum mempunyai label disebut dokumen uji. Penentuan kedekatan antara dokumen uji dan dokumen latih disebut kesamaan antar dokumen. Kesamaan dokumen uji dengan dokumen latih ini dapat dilakukan dengan menghitung jarak antara dokumen latih dan dokumen uji dengan menggunakan kesamaan cosines. Pada metode ini setiap *term* dinyatakan sebagai representasi dirnya sendiri secara unik sehingga kesamaan *cosines* akan dijalankan pada ruang ruang vektor yang besar. Dengan menerapkan LSI diharapkan akan diperoleh hasil pengklasifikasi yang lebih baik karena nois dan kata-kata yang semakna telah telah direpresentasi dengan satu *term.*

2. TINJAUAN PUSTAKA

2.1. Penelitian Terkait LSI

Penelitian yang memamfaatkan LSI dalam pengklasifikasian dokumen dapat dilihat pada penelitian yang diadakan oleh Cheng (Li and Park, 2007), klasifikasi dokumen dikonstruksi menggunakan jaringan saraf tiruan yang diintegrasikan dengan LSI. Hasil uji coba menunjukkan bahwa sistem pelatihan dengan LSI lebih cepat dibandingkan dengan model ruang vektor asal.

Penggunaan LSI pada klasifikasi dokumen saat ini dilaporkan oleh Liping (Jing et al., 2010). Pada penelitian ini diusulkan representasi suatu dokumen unit *term* semantik yang diidentifikasi dari informasi semantik implisit dan eksplisit. Semantik implisit diesktrak dari konten sintaksi dengan LSI sementara semantik eksplisit diperoleh dari sumber semantik ekternal yakni Wikipedia.

2.1.1. *Latent Semantic Indexing (LSI)*

LSI merupakan metode yang digunakan untuk mencari hubungan kata yang mempunyai makna atau sematik tersembunyi. Semantik tersembunyi ini dapat digunakan untuk mencari relasi antar kata berdasarkan makna. Dengan demikian LSI sangat bagus digunakan dalam hal sinonim, namun gagal untuk polisemi (Bassil and Semaan, 2012). Tujuan utama dari pengindekan LSI adalah untuk mengelompokkan *term-term* hasil ekstraksi dari dokumen latih dan dokumen uji berdasarkan kesamaan semantik antara *term*, sehingga *term-term* yang mempunyai kemiripan semantik dikumpulkan dalam satu kelompok. Pengindekan dengan LSI ini akan mengahasilkan kumpulan *term* yang telah tereduksi dari kumpulan *term* asal. Dengan demikian LSI akan mereduksi ruang vektor dengan membuat suatu subruang dari dimensi matrik yang bertujuan untuk menghapus *noise* dan *term* yang redundan. Ruang tereduksi tersebut menyajikan hubungan makna antara *term* dan dokumen.

LSI merupakan turunan dari teknik VSM yang dalam prosesnya memerlukan tiga tahap yang meliputi pembangunan *Term Document Matrix* (TDM), pembobotan, dan hasil perangkingan dokumen yang relevan berdasarkan similaritas. LSI akan mereduksi dimensi TDM dari pembobotan matriks kata dan dokumen dengan menggunakan *Singular Value Decomposition* (SVD)

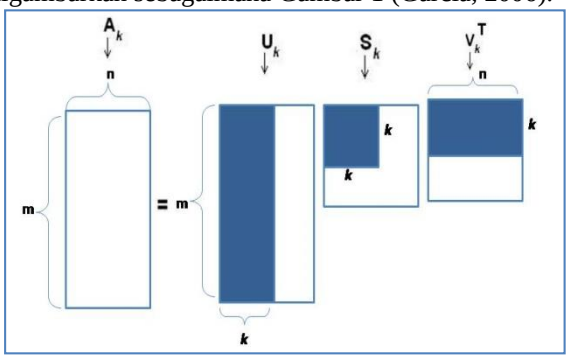
2.1.2. *Singular Value Decomposition* (SVD)

SVD merupakan sebuah model perkiraan yang digunakan untuk LSA (*Latent Semantic Analysis*), dimana komponen SVD dapat melakukan dekomposisi matriks sehingga didapatkan nilai dimensi yang lebih rendah (Peter et al., 2009). Dekomposisi SVD merupakan sebuah metode penulisan matriks dalam bentuk perkalian antara mariks diagonal yang berisi nilai-nilai singular dengan matriks yang berisi vektor-

vektor singular yang bersesuaian. Suatu matriks setelah dituliskan dalam matriks diagonal nilai singular dan matriks vektor singularnya dapat dilakukan reduksi *rank* pada matriks diagonal nilai singularnya. Reduksi *rank* beguna untuk mengurangi waktu komputasi suatu algoritma yang membutuhkan perkalian matriks.

Pemilihan nilai *k-rank* pada *Latent Semantic Indexing* (LSI) yang optimal merupakan hal yang sulit dipahami. Nilai optimal *k-rank* dipilih berdasarkan hasil eksekusi sekumpulan *query* dengan berbagai macam inputan *k-rank*. Hasil evaluasi paling tinggi dari nilai suatu *k-rank*, maka nilai *k-rank* tersebut merupakan nilai *k-rank* yang optimal (Kontostathis and Pottenger, 2006).

Pada SVD, *Term Document Matrix* (TDM) didekomposisi ke dalam 3 bentuk matriks yang digambarkan sebagaimana Gambar 1 (Garcia, 2006).



Gambar 1. Ilustrasi dari *Singular Value Decomposition* (SVD)

Dari ilustrasi Gambar 1 , dapat ditarik persamaan seperti berikut :

$$A = U.S.V^T \qquad (1)$$

dimana, *U* merupakan matriks *othonormal* dan barisnya sesuai dengan baris yang ada di matriks *A*. Matriks *S* merupakan matriks persegi berisi matriks diagonal yang tidak sama dengan 0, hanya terdapat nilai sepanjang diagonal matriks. Sedangkan matriks *V* merupakan matriks *othonormal* yang memiliki kolom yang sesuai dengan kolom pada matriks *A*, namun baris dari matriks *V* dibangun dari vektor *singular* (Ab Samat et al., n.d.).

Matriks *S* dihitung melalui prosedur-prosedur seperti berikut (Garcia, 2006) :

1. A^T dan $A.A^T$ dihitung
2. Mencari *eigenvalue* dari hasil perkalian matriks dan mengurutkan nilai dari yang terbesar ke yang terkecil. Hasil atau nilai dari matriks *S* adalah nonnegatif matriks, yang merupakan akar dari nilai pengurutan dan disebut dengan nilai *singular* dari *A*.
3. *S* dibangun dengan menempatkan nilai yang singular dimana nilai diurutkan dari yang terbesar ke yang terkecil pada setiap diagonalnya.

Dari persamaan 1 dapat dibentuk :

$$A.V = U.S.V^T.V = U.S$$
$$U^T A.V = S \qquad (2)$$

dimana :
A = matriks *A* yang dibangun dari TDM pembobotan ternormalisasi pada *corpus*.
V = matriks *V* hasil dekomposisi SVD matriks *A*
S = matriks singular hasil dekomposisi SVD matriks *A*
V^T = matriks *V transpose*
U = matriks *U* hasil dekomposisi SVD matriks *A*
U^T = matriks *U transpose* hasil dekomposisi SVD matriks *A*

4. *U* dan *V* adalah *ortoghonal*, dimana matriks orthogonal merupakan sebuah matriks yang jika dikalikan dengan *transpose*nya akan menghasilkan matriks identitas. Misalkan matriks *M* adalah *orthogonal* maka dapat ditulis bahwa $MM^T = M^TM = I$ =1. Perkalian *M* dan M^T bersifat komutatif(Garcia, 2006). Sehingga berdasarkan prosedur yang telah dijelaskan sebelumnya, nilai *eigenvalue* dapat dicari melalui persamaan berikut :

$$\left|A^T.A - cI\right| = 0 \qquad (3)$$

dimana :
$|A^T.A - cI|$ = de*term*inan nilai $A^T.A - cI$
A= matriks A : dari TDM pembobotan *corpus*
A^T = matriks A *transpose* dari TDM pembobotan *corpus*
c = merupakan variable nilai *eigen*
I = matriks indentitas

5. Nilai c merupakan nilai *eigen* yang akan dihasilkan oleh persamaan tersebut. Akar dari nilai *eigen* disebut dengan nilai *singular*. Nilai *singular* tersebut disusun berurutan dari nilai yang terbesar ke yang terkecil dan pada akhirnya membentuk matriks diagonal yang disebut dengan matriks *S*. Dari nilai *eigen* akan didapatkan *eigenvector* untuk membentuk matriks *V*, sesuai dengan persamaan berikut :

$$(A^T.A - cI)x = 0 \qquad (4)$$

dimana :
A= matriks *A* : dari TDM pembobotan *corpus*
A^T = matriks *A transpose* dari TDM pembobotan *corpus*
c = merupakan variable nilai *eigen*
I = matriks indentitas
x = variable *x*

6. Untuk mencari nilai dari matriks *U* maka digunakan persamaan berikut :

$$U = A.V.S^{-1} \qquad (5)$$

dimana :
U = hasil dari perkalian
A= matriks *A* : dari TDM pembobotan *corpus*
V = matriks *V* hasil dari nilai *eigenvector*
S^{-1} = *inverse* dari matriks singular

7. Proses dekomposisi dari reduksi dimensi SVD kemudian dapat digunakan untuk pengembangan sistem temu kembali. Untuk menghitung *query*

vektor dari SVD maka dapat dihitung dengan menggunakan persamaan berikut :

$$q' = q^T . U_k . S_k^{-1} \quad (6)$$

dimana :

q' = *query vector* representasi dari LSI

q^T=*transpose* TDM dari pembobotan ternormalisasi TF-IDF *query*

U_k = reduksi dimensi *k* dari matriks *U*

S_k^{-1} = *inverse* dari reduksi dimensi *k* matriks *S*

2.1.3. Algoritma LSI

Untuk melakukan LSI pada dokumen latih dan dokumen uji, dilakukan beberapa tahapan berikut :

Tahap 1 : Lakukan serangkaian praproses yang akan mengubah semua dokumen latih dan dokumen uji masing-masing menjadi matrik A dan B.

Tapah 2: Lakukan SVD pada matrik A menggunakan persamaan 1. Dengan pemilihan k-rank tertentu akan diperoleh U_k hasil reduksi k-rank.

Tahap 3 : Gunakan persamaan 6 untuk mendapatkan matrik tereduksi representasi dokumen latih. Dalam hal ini q digantikan dengan matrik B sehingga $B' = B^T . U_k . S_k^{-1}$.

2.1.4. Pengindekan dengan LSI

Proses pengindekan dengan LSI dilakukan dalam beberapa tahap sebagai berikut (Garcia, 2006):

Tahap 1. Gunakan algoritma LSI untuk mendapatkan matrik A' dan B' representasi masing-masing document latih dan dokumen uji tereduasi k-rank.

Tahap 2. Hitung *cosine* similaritas persamaan 7 antar dokumen latih dan dokumen uji hasil reduksi berdasarkan matrik tereduksi A' dan B', dalam hal ini X merepresentasikan dokumen uji dan d_j representasi dokumen latih ke j.

$$sim(X, d_j) = \frac{\sum_{j=1}^{m} X_j . d_{ij}}{\sqrt{\left(\sum_{j=1}^{m} X_j\right)^2} \cdot \sqrt{\left(\sum_{j=1}^{m} d_{ij}\right)^2}} \quad (7)$$

2.2. Metode KNN

Misalkan terdapat j kategori latih C_1, C_2, ...,C_j dari jumlah dari sampel latih N dan suatu dokumen uji Q. Setelah pre-prosesing, masing-masing dokumen akan menjadi vektor fitur representasi *term* unik berdimensi m. Selanjutnya akan dihitung kedekatan X dengan seluruh dokumen latih menggunakan cosine similaritas. Selanjutnya dokumen uji X akan ditentukan kategorinya berdasarkan k tetangga terdekat yang paling dominan kelasnya. Langkah-langkah untuk penerapan metode ini adalah sebagai berikut:

1. Membuat dokumen latih dan dari semua uji menjadi bentuk vektor fitur yang berdimensi sama.
2. Reduksi fitur berdimensi m dengan menggunakan LSI.
3. Hitung kesamaan antara dokumen X dengan semua dokumen latih menggunakan persamaan 6.
4. Memilih k sampel dengan menghitung probabilitas X ke masing-masing kategori menggunakan persamaan 7.

$$P(X, C_j) = \sum_{d_i \in KNN} SIM(X, d_i) . y(d_i, C_j) \quad (8)$$

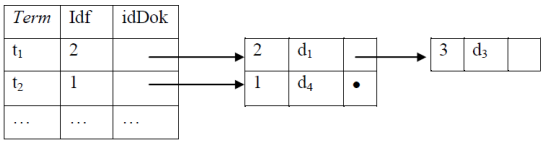
Dimana, $y(d_i, C_j)$ adalah fungsi attribut kategori yang memenuhi persamaan 9

$$y(d_i, C_j) = \begin{cases} 1, d_i \in C_j \\ 0, d_i \notin C_j \end{cases} \quad (9)$$

Uji dokumen X untuk mengetahui kategorinya dengan melihat $P(X,C_j)$ terbesar.

2.3 *Inverted Index*

Inverted Index merupakan struktur data yang digunakan untuk mengindek keseluruhan kata unik pada semua dokumen korpus (Zelikovitz and Hirsh, 2001). Representasi struktur data inverted indeks dapat digambarkan sebagaimana pada Gambar 2.



Gambar 2. Ilustrasi Struktur Data Inverted Index

2.4. Pembobotan TF-IDF Ternormalisasi

TF-IDF dengan normalisasi frekuensi dapat meningkatkan proses pembobotan dari kata. Fungsi normalisasi untuk mengurangi efek dari panjang dokumen dengan menggunakan persamaan 10 sebagai berikut (Bassil and Semaan, 2012) :

$$f_{i,j} = \frac{tf_{i,j}}{\max tf_{i,j}} \quad (10)$$

dimana

fi,j = frekuensi ternormalisasi

tfi,j = frekuensi kata i pada dokumen j

max tfi,j = frekuensi maksimum kata i pada dokumen j

Untuk normalisasi frekuensi dalam query diberikan rumus :

$$f_{Q,i} = 0{,}5 + 0.5 * \frac{tf_{Q,i}}{\max tf_{Q,i}} \quad (11)$$

dimana

fi,j = frekuensi ternormalisasi

tfi,j = frekuensi kata i pada dokumen j

max tfi,j = frekuensi maksimum kata i pada dokumen j

Sehingga pembobotan TF-IDF pada kata i dan dokumen j dapat ditulis sebagai berikut:

$$W_{i,j} = \frac{tf_{i,j}}{\max f_{i,j}} \times \log_2\left(\frac{D}{df_i}\right) \quad (12)$$

dimana :

$W_{i,j}$ = bobot kata i pada dokumen j

$f_{i,j}$ = frekuensi ternormalisasi

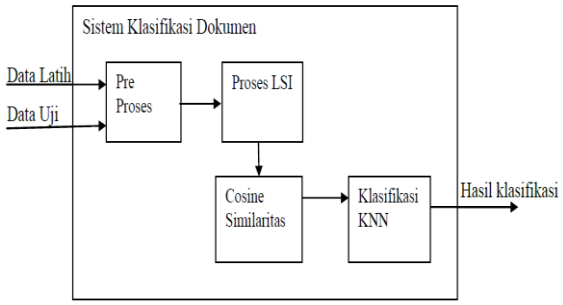
$tf_{i,j}$ = frekuensi kata i pada dokumen j

max $tf_{i,j}$ = frekuensi maksimum kata i pada dokumen j
D = banyaknya dokumen yang diinputkan/ banyaknya dokumen dalam *corpus*
df_i = banyaknya dokumen yang mengandung kata *i*

3. ARSITEKTUR SISTEM

Arsitektur Sistem

Arsitektur sistem secara keseluruhan terdiri dari serangkaian tahap sebagaimana pada Gambar 3 berikut.



Gambar 3. Arsitektur Sistem

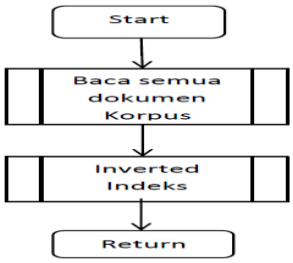
1. Tahap pertama yang dilakukan adalah melakukan pre proses terhadap data latih maupun data uji. Target akhir dari pre proses ini adalah merepresentasikan dokumen ke dalam format angka yang siap untuk dioperasikan pada fase-fase selanjutnya. Tahap praproses ini meliputi tokenisasi yakni memecah masing-masing dokumen ke dalam *term-term*, hasil ekstraksi *term-term* dari seluruh dokumen dihilangkan penghilangan kata *stop list* (kumpulan kata yang tidak penting) sehingga hanya akan terbentuntuk kumpulan *term-term* penting. Dari seluruh kata yang dianggap penting dilakukan proses *stemming* untuk mendapatkan akar kata dari masing-masing *term*. Pada penelitian ini akan diadopsi algoritma *stemming Porter* yang telah diadaptasi untuk bahasa Indonesia oleh Tala [4]. Kumpulan kata dasar yang diperoleh selanjutnya adakan dijadikan tem unik. Akhir dari pre proses ini semua *term* akan dimasukkan ke dalam struktur data *inverted Index* sebagai mana pada Gambar 2.
2. Mengektrak bobot masing-masing *term* menggunakan tf-idf baik dari dokumen latih maupun dokumen uji ke dalam representasi matrik masing-masing A dan B.
3. Lakukan reduksi fitur dengan menggunakan LSI
4. Hitung kesamaan antara masing-masing dokumen uji dan dokumen latih menggunakan persamaan 6.
5. Hitung klasifikasi masing-masing dokumen uji menggunakan KNN.
6. Evaluasi hasil menggunakan metric presisi, recal dan F-Metrik.

4.1.2. Pengembangan Sistem

Berdasarkan rancangan sistem di atas, selanjutnya dikembangkan sistem pengklasifikasian dokumen dengan metode KNN berbasis LSI. Sistem dikembangkan menggunakan bahasa pemrograman java yang dijalankan *platform window* 7 dengan memamfaatkan *library* Jama untuk proses SVD.

4.1.3. Rancangan tahap praprosesing

Tahap praprosesing tujuan utamanya mengubah representasi data dokumen teks menjadi representasi numerik yang siap untuk diolah lebih lanjut. Pada tahap ini secara umum dapat digambarkan menggunakan flowchart Gambar 4 berikut :



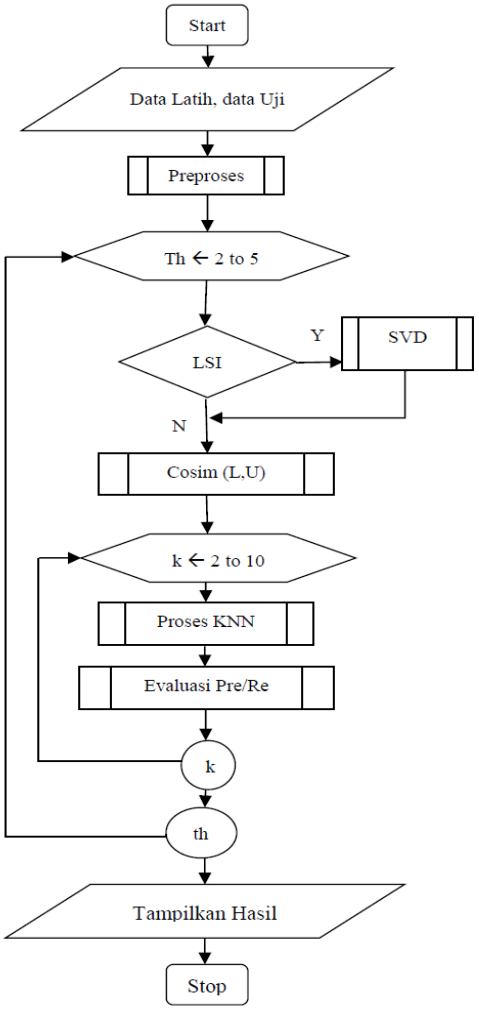
Gambar 4 Tahap *Praprosesing*

Modul baca semua dokumen korpus diawali dengan membaca direktori data korpus dan menyimpan dalam dua array dLat dan dUji. Kedua array ini menjadi parameter masukan untuk memanggil konstruktur class *Inverted indeks*. Selanjutnya pada class inverted indeklah akan dilakukan semua tahapan praprosesing mulai dari tokenisai, filterisasi, steming serta penghitungan frekuensi kemunculan *term* pada masing-masing dokumen.

4. Skenario Evaluasi Sistem

Pengaruh pengindekan dengan LSI dihitung berdasarkan presisi, recall dan FMeasure-nya pada setiap kategori. Evaluasi system meliputi beberapa scenario sebagai berikut :

a. Secara umum akan dievaluasi pengaruh LSI pada algoritma KNN sehingga terdapat KNN biasa dan KNN-LSI.
b. Pada setiap uji coba KNN dan KNN-LSI masing-masing akan akan dievaluasi pada beberapa nilai batas frekuensi kata (*threshould*) mulai dari 2 sampai 5. Tujuan dari evaluasi ini untuk mengetahui sensitifitas threshould pada kedua metode KNN dan KNN-LSI.
c. Evaluasi pada setiap *threshould* akan dievaluasi pada berbagai nilai k mulai dari 2 sampai 10 sebagai parameter dari KNN.
d. Seluruh uji coba a sampai c juaga akan dievaluasi pada praproses *stemming* dan non *stemming* untuk mengetahui sensitifitas *stemming* pada kedua metode. Skenario sistem secara keseluruhan dapat digambarkan sabagaimana *flowchart* pada Gambar 5.



Gambar 5. *Flowchart* Skenario Uji coba sistem

Tahap praproses pada Gambar 5 dilakukan untuk mengubah semua dokumen latih dan dokumen uji yang berupa teks menjadi representasi angka yang siap diolah pada proses berikutnya melalui serangkaian sub proses sebagaimana dijelaskan pada tahap pertama dari arsitektur system pada penjelasan Gambar 3. Hasil dari praproses adalah berupa matrik L dan U masing-masing merepresentasikan data latih dan data uji dengan ukuran baris antara keduanya sama yakni sesuai dengan jumlah *term* unik yang memenuhi *threshould* th hasil praproses yang telah disimpan pada inverted indeks. Th disini dimaksudkan besarnya frekuensi minimal dari masing-masing *term* dimulai dari 2, sampai dengan 5. Sedangkan ukuran kolom masing-masing tergantung jumlah dokumen latih untuk matrik L dan jumlah dokumen uji untuk matrik U. Jika dipilih LSI maka akan dipanggil modul SVD. Selanjutnya hasil dari SVD atau yang tanpa proses ini kedua matrik L dan U akan dicari kesamaannya menggunakan *cosine similariry*. Matrik representasi kesamaan antara kedua matrik L dan U selanjutnya digunakan oleh algoritma KNN yang diuji pada berbagai kondisi nilai k. Looping pertama dimaksudkan untuk mengetahui sejauhmana pengaruh *threshould* frekuensi *term* unik terhadap hasil sistem, baik dengan metode KNN-LSI atau KNN biasa. Selain itu akan dievaluasi untuk masing-masing *threshould*-nya pada nilai k berapakah yang paling optimal.

Perbedaan antara uji coba KNN dan KNN-LSI terletak pada pemilihan proses LSI. Jika tidak dipilih proses LSI maka kedua matrik L dan U yang merepresentasikan masing-masing data latih dan data uji tidak mengalami proses dekomposisi SVD. Sehingga kedua matrik ini akan menjadi masukan untuk menghitung kesamaan antara data latih dan data uji melalui modul Cosim(L,U). Sebaliknya, jika dipilih LSI maka sebelum dihitung kesamaan antara kedua data tersebut terlebih dahulu dilakukan proses SVD. Setelah proses SVD matrik L dan U akan mengalami reduksi sesuai dengan pemilihan k-rank. Selanjutnya kedua matrik hasil reduksi ini akan dihitung kesamaannya menggunakan Cosim(L,U).

Hasil dari masing-masing metode selanjutnya akan dievaluasi untuk menghitung presesi, recall dan f1-nya. Hasil evaluasi inilah yang natinya akan dijadikan bahan argumentasi kinerja kedua metode.

4.2. Rancangan Data

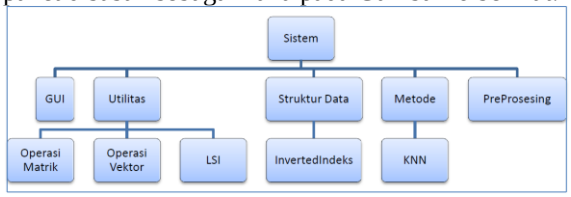
Dokumen latih dan dokumen uji pada penelian ini berasal dari berita berbahasa Indonesia yang diambil dari internet. Selanjutnya masing-masing dokumen yang berasal dari kategori yang sama dikelompokkan dalam satu direktori yang diberi label sama dengan kategorinya. Semua data diorganisir dalam dua kelompok besar yakni sebagai kelompok data latih dan kelompok yang lain sebagai data uji.

5. HASIL DAN PEMBAHASAN

5.1. Lingkungan sistem

Sistem dikembangkan dengan menggunakan bahasa pemrograman java dengan jdk jdk1.7.0_79 dengan IDE Neatbean 8.1 yang dijalankan pada lingkungan Windows 7 pada Laptop berprosesor intel i3 dengan RAM 4 G.

Sistem dikembangkan dengan bahasa pemrograman java berbasis objek oriented yang disusun dalam class-class dan paket-paket. Organisasi paket-paket disusun sebagaimana pada Gambar 6 berikut.



Gambar 6. Stukrut paket-paket program sistem yang dikembangkan

Selain paket-paket diatas didukung oleh library jama-1.0.3 yang merupakan paket untuk semua operasi dasar matrik.

5.2. Evaluasi kinerja sistem

Uji coba dilakukan pada sejumlah data sebanyak 523 dokumen yang terdiri dari 8 kategori. Setelah dipilih secara acak dari masing-masing kategori maka sebaran data dapat digambarkan sebagaimana tabel 1.

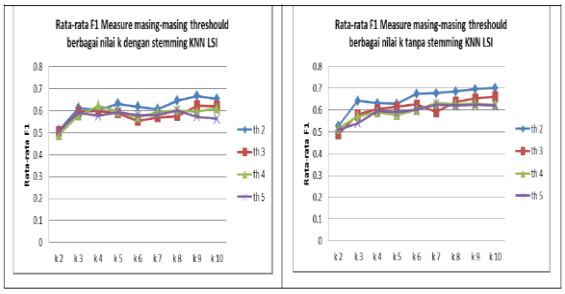
Tabel 1. Distribusi data latih dan data uji

	Dt latih	Dt Uji
Edukasi	42	18
Ekonomi	32	29
Kesehatan	39	10
Olahraga	36	29
Otomotif	40	26
Politik	48	23
Sains	48	14
Teknologi	47	42
Total	332	191

Evaluasi sistem keseluruhan dijalankan sebagaimana flowchart gambar 5 Selain diukur kinerja sistem pada berbagai situasi stemming atau non stemming dan antara KNN biasa dengan KNN LSI, sistem juga diukur waktu eksekusi masing-masing skenario. Penghitungan waktu eksekusi dilakukan dengan mencatat waktu awal eksekusi dan waktu akhirv pada masing-masing metode.

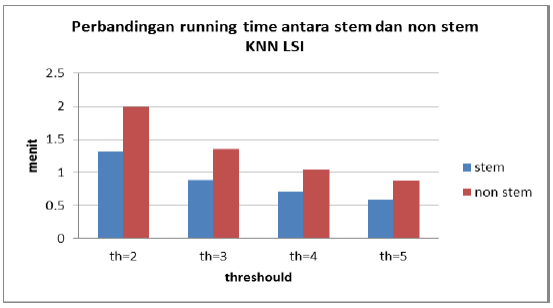
5.2.1. Rata-rata F1-Measure masing-masing *threshould* pada berbagai nilai k

Untuk mengetahui kinerja sistem secara keseluruhan, maka dilakukan perhitungan rata-rata F1 dari semua kategori pada berbagai threshoud baik pada KNN LSI maunpun KNN biasa. Pada evaluasi ini sekaligus diperlihatkan sensitifitas penggunaan *stemming* dan tidak pada kedua metode. Perbandingan hasil rata-rata F1 measure pada KNN LSI pada kedua konsidi *stemming* dan tidak dipelihatkan sebagaimana pada Gambar 7.



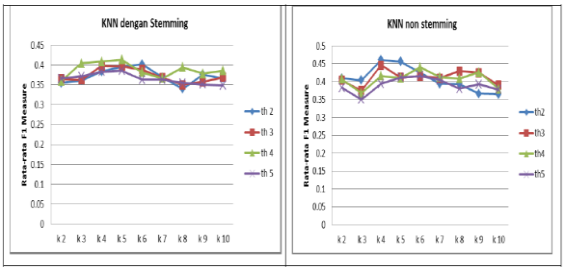
Gambar 7. Perbandingan rata-rata F1 measure masing-masing *threshould* pada berbagai nilai k antara non *stemming* dan *stemming* pada KNN LSI

Dari Gambar 11 dapat diketahui bahwa kinerja terbaik untuk *stemming* LSI ditunjukkan paa *threshould* 2 dan nilai k=9 dengan nilai F1 measure rata 0.6682, demikian juga pada LSI non *stemming* kinerja terbaik terletak pada *threshould* 2 pada k=10 dengan nilai F1 measure rata-rata 0.701775. Dengan demikian kinerja terbaik untuk KNN LSI dicapai tanpa menggunakan *stemming*. Namun demikian berbanding terbalik ketika mempertimbangkan sisi waktu running sebagaimana ditunjukkan pada gambar 8.



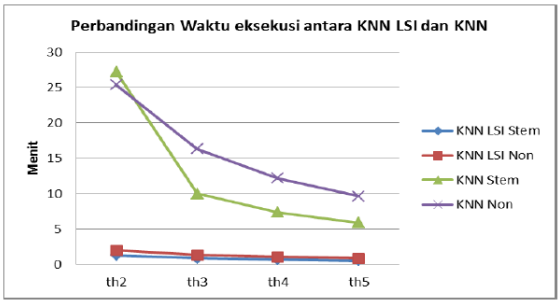
Gambar 8. Perbandingan waktu running antara stem dan non stem pada KNN LSI

Sedangkan kinerja sistem dengan KNN tanpa LSI jauh lebih jelek dibandingkan dengan KNN LSI baik dengan *stemming* maupun tanpa *stemming* sebagaimana ditunjukkan pada Gambar 9.



Gambar 9. Perbandingan F1 measure antara KNN dengan steming dan KNN tanpa *stemming*

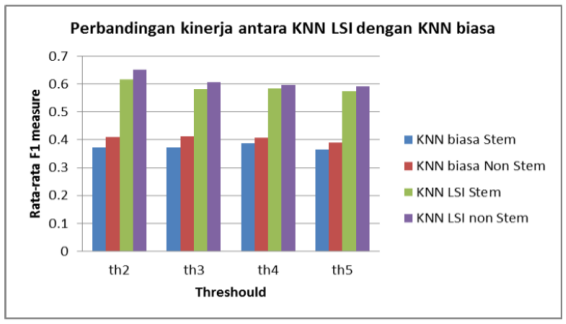
Rata-rata F1 measure pada KNN non LSI hanya dicapai maksimal 0.4611875 yakni pada th=2 tanpa *stemming*. Selain itu perbadingan waktu eksekusi antara KNN LSI dan KNN biasa menunjukkan perbedaan yang signifikan sebagimana disajikan pada Gambar 10 berikut.



Gambar 3. Perbedaan waktu eksekusi antara KNN LSI dan KNN biasa

5.2.2. Perbandingan kinerja KNN non LSI dengan KNN LSI

Evaluasi terakhir dari sistem adalah membandingkan kinerja klasifikasi dokumen dengan KNN LSI dengan KNN biasa. Evaluasi dihitung berdasarkan rata-rata F1 measure dari masing-masing *threshould*. Hasil evaluasi menunjukkan bahwa KNN LSI kinerjamya lebih baik dari KNN biasa sebagaimana ditunjukkan pada Gambar 1.



Gambar 4. Perbandingan antara KNN non LSI dan KNN LSI

5.3. Analisa dan Pembahasan

Pada bagian ini akan dibahas analisa hasil uji coba sistem. Sebagaimana target yang telah ditetapkan pada bagian tujuan penelitian ini yakni membandingkan pengaruh ektraksi fitur dengan LSI dan non LSI pada kinerja dari sisi waktu maupun hasil pengklasifikasi dokumen berbahasa Indoneisa dengan metode KNN baik dengan *stemming* maupun non *stemming* pada berbagai perlakukan *threshould* dan nilai k.

Evaluasi diawali dengan melihat pengaruh *stemming* terhadap jumlah dimensi *term*. Selanjutnya kecepatan waktu eksekusi menjadi bahan pertimbangan untuk pengujian selanjutnya. Waktu eksekusi sangat dipengaruhi oleh besarnya dimensi matrik data latih dan data uji. Hal ini disebabkan karena dalam proses penghitungan kesamaan antara kedua kelompok dokumen latih dan kelompok dokumen uji haruslah dilakukan dengan cara melakukan perkalian matrik antara keduanya. Sedangkan perkalian matrik mempunyai kompleksitas $O(n^3)$ yakni $U^T_{u,t}$ x $L_{t,l}$ dimensi yang dibutuhkan u*t*l, dimana u menyatakan banyaknya data uji, t menyatakan banyaknya *term* unik dan l menyatakan banyaknya data latih. Kondisi yang mungkin untuk direduksi adalah t. Untuk itulah dilakukan pemilihan *term* berdasarkan frekuensi minimal yakni 2, 3, 4 dan 5 baik dengan *stemming* maupun non *stemming*. Selanjutnya masing-masing skenario pemilihan fitur ini diujicobakan pada KNN biasa dan KNN LSI. Hasil evaluasi menunjukkan kinerja waktu yang sangat signifikan antara KNN LSI dan KNN biasa, akan tetapi tidak begitu signifikan pada KNN LSI antara *stemming* non stemming *stemming*. Hasil terbaik dari sisi waktu eksekusi dicapai ketika menggunakan KNN LSI dengan *stemming* pada *thresould* 5. Hal ini sesuai sesuai dengan fakta bahwa dengan *stemming* fitur akan tereduksi dan dampaknya pada reduksi dimensi matrik baik latih maupun testing. Demikian pula jika dibandingkan kinerja hasil pada KNN biasa dengan KNN LSI hasilnya jauh lebih metode KNN LSI. Namun demikian jika dibandingkan kinerja hasil pada KNN LSI antara *stemming* dan non *stemming*, hasilnya lebih baik pada non *stemming* pada *threshould* 2. Dengan demikian dapat ditarik suatu kesimpulan bahwa dengan KNN LSI non *stemming* menghasilkan kinerja terbaik dari sisi hasil akan tetapi dari sisi waktu KNS LSI dengan *stemming* pada *threshould* 5 menunjukkan kinerja terbaik. Kombinasi ekstraksi fitur dengan *threshould* = 2, non *stemming* dan dekomposisi matrik dengan LSI akan menghasilkan reduksi fitur yang sangat signifikan dan menghilangkan bias *sinonemi* pada fitur sebagai akibat penggunaan LSI. Karena bias *sinonemi* berkurang maka kesamaan antar dokumen semakin meningkat sehingga dampaknya terlihat pada kinerja hasil yang terbaik dari semua situasi.

6. KESIMPULAN DAN SARAN

Kesimpulan

Berdasarkan hasil dan pembahasan sebagaimana pada bab 5 maka dapat ditarik beberapa kesimpulan : klasifikasi dokumen berbahasa Indonesia dengan menggunakan KNN LSI lebih baik dari KNN non LSI, namun demikian kinerja hasil terbaik ditunjukkan pada KNN LSI non *stemming* pada *threshould* 2 sedangkan kinerja terbaik dari sisi waktu dicapai ketika *sistem* menggunakan KNN LSI dengan *stemming* pada *threshould* 5.

7. DAFTAR PUSTAKA

Ab Samat, N., Murad, M.A.A., Atan, R., Abdullah, M.T., n.d. Categorization of Malay Documents using Latent Semantic Indexing.

Bassil, Y., Semaan, P., 2012. Semantic-Sensitive Web Information Retrieval Model for HTML Documents. ArXiv Prepr. ArXiv12040186.

Garcia, E., 2006. Latent Semantic Indexing (LSI) A Fast Track Tutorial. September.

Jing, L., Yun, J., Yu, J., Huang, H., 2010. Text Clustering via Term Semantic Units. IEEE, pp. 417–420. doi:10.1109/WI-IAT.2010.23

Kontostathis, A., Pottenger, W.M., 2006. A framework for understanding Latent Semantic Indexing (LSI) performance. Inf. Process. Manag. 42, 56–73.

Li, C.H., Park, S.C., 2007. Artificial Neural Network for Document Classification Using Latent Semantic Indexing. IEEE, pp. 17–21. doi:10.1109/ISITC.2007.69

Peter, R., Shivapratap, G., Divya, G., Soman, K.P., 2009. Evaluation of svd and nmf methods for latent semantic analysis. Int. J. Recent Trends Eng. 1.

Silva, I.R., Souza, J.N., Santos, K.S., 2004. Dependence among terms in vector space model, in: Database Engineering and Applications Symposium, 2004. IDEAS'04. Proceedings. International. IEEE, pp. 97–102.

Wu, H., Gunopulos, D., 2002. Evaluating the utility of statistical phrases and latent semantic indexing for text classification, in: Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on. IEEE, pp. 713–716.

Yang, Y., 1995. Noise reduction in a statistical approach to text categorization, in: Proceedings of the 18th Annual International

ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 256–263.

Zelikovitz, S., Hirsh, H., 2001. Using LSI for text classification in the presence of background text, in: Proceedings of the Tenth International Conference on Information and Knowledge Management. ACM, pp. 113–118.